# The logic route to strong AI

## Alibaba HKAI Lab 2020 presentation

Heelal team

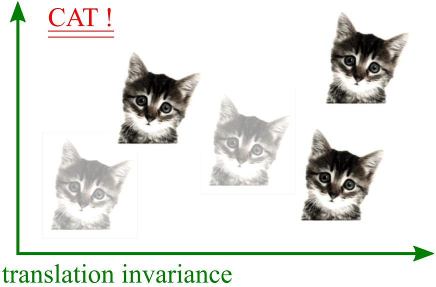
July 24, 2020

# Table of contents

1 CNN
2 Symmetry and inductive bias
3 Richard Sutton
4
5 Structure of logic
6 Symmetric neural networks
8 Application: knowledge graphs
9 Application: logicalization of BERT
10 Application: content-addressable long-term memory
11

# CNN

-



(1)

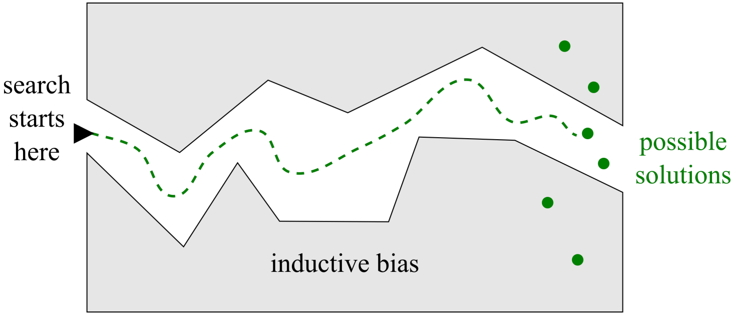
- Convolution

$$(T_x \circ f) * g = T_x \circ (f * g) \tag{2}$$

- Yann LeCun CNN

# Symmetry and inductive bias

-
- (inductive bias)

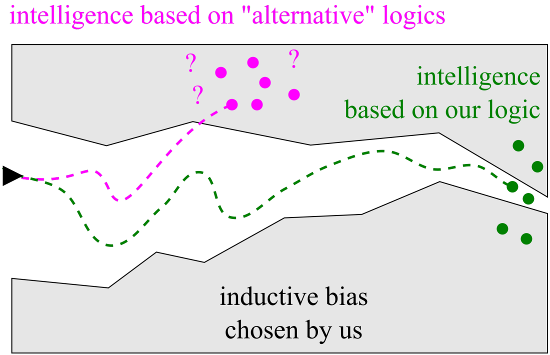


(3)

- (intractable)

# Richard Sutton

- Richard Sutton strong AI
-



(4)

- AGI

-
- (logical form)
- models model
-

(5)

-
- model
-

# Structure of logic

- strong AI
- inductive bias the solution to strong AI
- symmetry (commutativity, or permutation invariance):

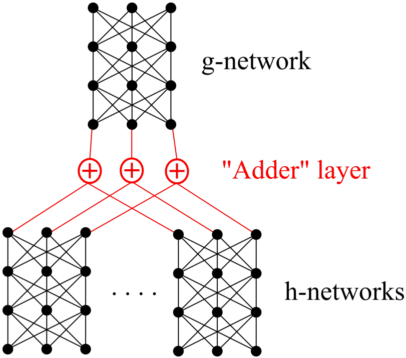
$$\begin{aligned} A \wedge B &\equiv B \wedge A \\ \wedge &\equiv \wedge \end{aligned} \tag{6}$$

-
- (mental state) (propositions)

# Symmetric neural networks

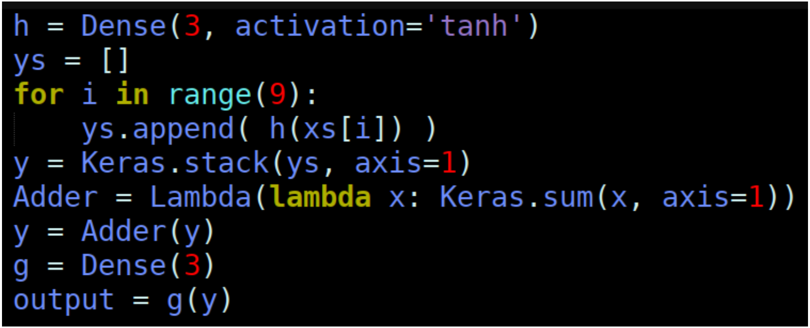
- Permutation invariance can be handled by symmetric neural networks
- I wasted 2 years trying to solve this problem, and then find out that it has been solved 3 years ago: [PointNet 2017] and [DeepSets 2017] and their mastery of mathematics is significantly above me!
- Any symmetric function can be represented by the following form (a special case of the Kolmogorov-Arnold representation of functions):

$$f(x, y, ...) = g(h(x) + h(y) + ...) \tag{7}$$



(8)

- Sym NN gives a powerful boost in efficiency $\propto n!$ where $n = \#$inputs
- The code for Sym NN is just a few lines of Tensorflow:



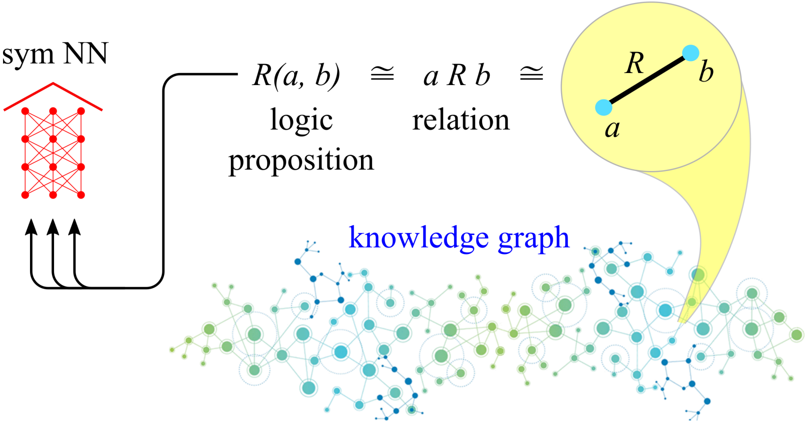
```
h = Dense(3, activation='tanh')
ys = []
for i in range(9):
    ys.append( h(xs[i]) )
y = Keras.stack(ys, axis=1)
Adder = Lambda(lambda x: Keras.sum(x, axis=1))
y = Adder(y)
g = Dense(3)
output = g(y)
```

(9)

- Very easy to adopt this to existing models such as BERT and reinforcement learning
- I have successfully tested it on the game of TicTacToe

# Application: knowledge graphs

- One cannot feed a knowledge graph directly into an NN, as its input must be embedded in vector space. A solution is to break the graph into edges, where each edge is equivalent to a relation or proposition. One could say that graphs are isomorphic to logic



sym NN

$$R(a, b) \cong a\,R\,b \cong \tag{10}$$

logic proposition

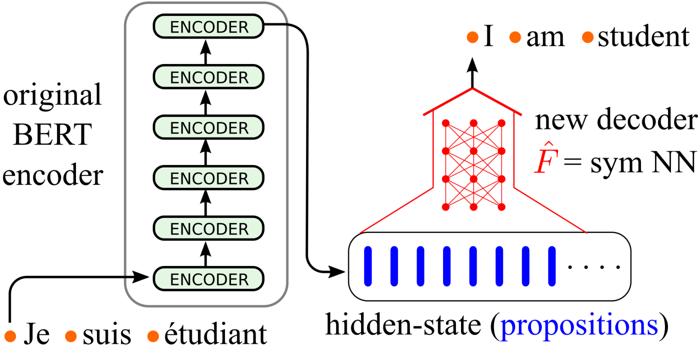
relation

knowledge graph

- Since edges are invariant under permutations, it appears that symmetric NNs are required to process them

# Application: logicalization of BERT

- Similarly, we can convert BERT's hidden state into a set of propositions, by replacing the original decoder with a sym NN:

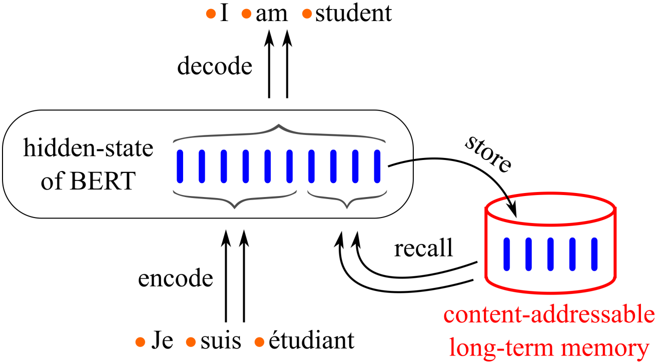


(11)

- The original encoder can be retained. As the decoder imposes symmetry on the hidden state, error propagation is expected to cause its representation to change
- Of course, this remains to be proven by experiment ☺️

# Application: content-addressable long-term memory

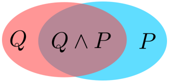
- The original BERT hidden state lacked a logical structure and it was not clear what exactly it contains. After logicalization, propositions inside BERT can be stored into long-term memory:



(12)

- These kind of systems are very close to strong AI, and it depends crucially on logicalization
- The content-addressable memory idea came from Alex Graves *et al*'s Neural Turing Machine [2014]

- AI
- (geometry) (topology)
- Venn diagrams



(13)

(predicates) base set (fibration), $\begin{tikzcd}\mathbb{E} \arrow[d, "p"] \\ \mathbb{B}\end{tikzcd}$

- Curry-Howard isomorphism

$$\begin{aligned} &\Leftrightarrow \quad \text{type theory} \qquad (14) \\ &\Leftrightarrow \quad \text{programs} \end{aligned}$$

-

- BERT word embedding concatenation

$$\cdot \quad \cdot \quad \xrightarrow{BERT} \quad \cdot \quad \cdot \tag{15}$$

$$\forall x.\ \mathsf{Human}(x) \rightarrow \mathsf{Mortal}(x) \tag{16}$$

Curry-Howard

- commutativity predicates
- “ ” topos HoTT (homotopy type theory)
-

# References

[1] Alex Graves, Greg Wayne, and Ivo Danihelka. "Neural Turing Machines". In: *CoRR* abs/1410.5401 (2014). arXiv: 1410.5401. URL: http://arxiv.org/abs/1410.5401.

[2] Qi et al. "Pointnet: Deep Learning on Point Sets for 3D Classification and Segmentation". In: *CVPR* (2017). https://arxiv.org/abs/1612.00593.

[3] Zaheer et al. "Deep sets". In: *Advances in Neural Information Processing Systems* 30 (2017), pp. 3391–3401.

**Illustration credits:**

- Translation invariance, from Udacity Course 730, Deep Learning (L3 Convolutional Neural Networks ▷ Convolutional Networks)
- Étale space, from Topoi – The Categorical Analysis of Logic [Goldblatt 2006]